



A Sierra Monitor Company

Driver Manual
(Supplement to the FieldServer Instruction Manual)

FS-8704-03 Modbus TCP

APPLICABILITY & EFFECTIVITY

Effective for all systems manufactured after August 2008

Driver Version:	1.01
Document Revision:	7

TABLE OF CONTENTS

1. MODBUS/TCP DESCRIPTION 3

2. DRIVER SCOPE OF SUPPLY..... 4

 2.1. Supplied by FieldServer Technologies for this Driver 4

 2.2. Provided by Supplier of 3rd Party Equipment 4

3. HARDWARE CONNECTIONS 4

4. CONFIGURING THE FIELDSEVER AS A MODBUS/TCP CLIENT 5

 4.1. Data Arrays 5

 4.2. Client Side Connection Descriptors 5

 4.3. Client Side Node Descriptors 6

 4.4. Client Side Map Descriptors..... 7

 4.4.1. *FieldServer Related Map Descriptor Parameters* 7

 4.4.2. *Driver Related Map Descriptor Parameters* 7

 4.4.3. *Timing Parameters* 7

 4.4.4. *Map Descriptor Example* 8

5. CONFIGURING THE FIELDSEVER AS A MODBUS/TCP SERVER 9

 5.1. Server Side Connection Descriptors 9

 5.2. Server Side Node Descriptors 9

 5.3. Server Side Map Descriptors 10

 5.3.1. *FieldServer Specific Map Descriptor Parameters* 10

 5.3.2. *Driver Specific Map Descriptor Parameters* 11

 5.3.3. *Map Descriptor Examples* 12

APPENDIX A. ADVANCED TOPICS..... 13

 Appendix A.1. Data Types 13

 Appendix A.2. Single Writes 14

 Appendix A.3. Read/write Operation 14

APPENDIX B. DRIVER ERROR MESSAGES 15

1. Modbus/TCP Description

The Modbus TCP driver allows the FieldServer to transfer data to and from devices over Ethernet using Modbus/TCP protocol. The driver was based on Modbus Application Protocol Specification V1.1a from Modbus-IDA. The FieldServer can emulate either a Server or Client. Up to 200 simultaneous Modbus connections are possible.

The information that follows describes how to expand upon the factory defaults provided in the configuration files included with the FieldServer.

There are various register mapping models being followed by various vendors
To cover all these models FieldServer uses the following three Models

- **Modicon_5digit** – Use this format where addresses are defined in 0xxxx, 1xxxx, 3xxxx or 4xxxx format. A maximum of 9999 registers can be mapped of each type. This is FieldServer driver's default format.
- **ADU** –Application Data Unit address. Use this format where addresses of each type are defined in the range 1-65536
- **PDU** –Protocol Data unit address. Use this format where addresses of each type are defined in the range 0-65535.

The key difference between ADU and PDU is for example if Address_Type is ADU and address is 1, the driver will poll for register 0. If Address_Type is PDU, the driver will poll for address 1.

Note 1: If vendor document shows addresses in extended Modicon (i.e. 6 digit) format like 4xxxxx then consider these addresses as xxxxx (just omit the first digit) and use either ADU or PDU

Note 2: The purpose of providing 3 different ways of addressing the Modbus registers is to allow the user to choose the addressing system most compatible with the address list being used. At the protocol level, the same protocol specification is used for all three with the exception of the limited address range for Modicon_5digit.

4. Configuring the FieldServer as a Modbus/TCP Client

Refer to section 4.1 of the Instruction Manual for a description of the operation principle of the FieldServer. The following tables describe parameters that need to be filled out in the configuration file. For convenience, a few example parameters already exist in the supplied configuration files.

Note that * indicates an optional parameter, with the bold legal value being the default.

4.1. Data Arrays

Section Title		
Data_Arrays		
Column Title	Function	Legal Values
Data_Array_Name	Provide name for Data Array	Up to 15 alphanumeric characters
Data_Format	Provide data format	UINT 16, UINT 32, SINT 16, SINT 32, BIT, FLOAT, Packed_Bit
Data_Array_Length	Number of Data Objects	1 – 255

Example:

```
// Data Arrays

Data_Arrays
Data_Array_Name,          Data_Format,          Data_Array_Length
DA_AI_01,                 Float,                200
DA_AO_01,                 Float,                200
DA_DI_01,                 Bit,                  200
DA_DO_01,                 Bit,                  200
```

4.2. Client Side Connection Descriptors

Section Title		
Connections		
Column Title	Function	Legal Values
Adapter	Specify which adapter this protocol uses	N1
Protocol	Specify protocol used	Modbus/TCP
Poll Delay*	Time interval between polls	0-32000 s, 0.05 s.

Example:

```
// Client Side Connections

Connections
Adapter,          Protocol,          Poll_Delay
N1,               Modbus/TCP,      0.100s
```

4.3. Client Side Node Descriptors

Section Title		
Nodes		
Column Title	Function	Legal Values
Node_Name	Provide name for Node	Up to 32 alphanumeric characters
Node_ID	Station Address of Remote Server Node	1 – 255
Protocol	Specify protocol used	Modbus/TCP
Adapter	Specify which adapter this protocol uses	N1
IP_address	IP address of client PLC	Valid IP address, e.g. 192.168.1.13
Node_Type ³	Set to Block_Mode if Remote Server Node (RSN) only supports Write Multiple – FC16 & FC15, and does not support FC05 or FC06	Block_Mode
Address_Type ⁴	Specify Register Mapping Model	ADU,PDU, -, Modicon_5digit

Example

```
// Client Side Nodes
// For new devices where 65536 registers could be available in each memory area
Nodes
Node_Name,      Node_ID,   Protocol,      Adapter,   Address_Type   IP_Address
Modbus device 1, 1,         Modbus/TCP,   N1,       ADU            192.168.1.172
Modbus device 2, 2,         Modbus/TCP,   N1,       PDU            192.168.1.172
// For devices where only 9999 registers could be available in each memory area
Nodes
Node_Name,      Node_ID,   Protocol,      Adapter
Modbus device 3, 3,         Modbus/TCP,   N1
```

³ If this parameter is not specified the default function codes will be FC 05 (Single_Coil) and FC 06 (Single_Register). Refer to Appendix A.3 for more information.

⁴ **Optional for Modicon 5 digit devices**

4.4. Client Side Map Descriptors

4.4.1. FieldServer Related Map Descriptor Parameters

Column Title	Function	Legal Values
Map_Descriptor_Name	Name of this Map Descriptor	Up to 32 alphanumeric characters
Data_Array_Name	Name of Data Array where data is to be stored in the FieldServer	One of the Data Array names from "Data Array" section above
Data_Array_Offset	Starting location in Data Array	0 to maximum specified in "Data Array" section above
Function	Function of Client Map Descriptor	Rdbc, Wrbc, Wrbcx or passive

4.4.2. Driver Related Map Descriptor Parameters

Column Title	Function	Legal Values
Node_Name	Name of Node to fetch data from	One of the Node names specified in "Client Node Descriptor" above
Data_Type ⁵	Specify memory area	Address_Type = ADU Coil, Discrete_Input, Input_Register, Holding_Register, Single_Coil, Single_Register
		Address_Type = PDU FC01, FC02, FC03, FC04, FC05, FC06, FC15, FC16
		Address_Type = Modicon_5digit - (Dash), Single_Register, Single_Coil
		Address_Type = ADU 1-65536
		Address_Type = PDU 0-65535
Address	Starting address of read block	Address_Type = Modicon_5digit 40001, 30001, etc
		Address_Type = ADU 1-65536
		Address_Type = PDU 0-65535
Length	Length in items to record from PLC	1 – 125 (depending on type)
Data_Array_Low_Scale*	Scaling zero in Data Array	Any signed 32 bit integer in the range: -2,147,483,648 to 2,147,483,647. 0
Data_Array_High_Scale*	Scaling max in Data Array	Any signed 32 bit integer in the range: -2,147,483,648 to 2,147,483,647. 100
Node_Low_Scale*	Scaling zero in Connected Node	Any signed 32 bit integer in the range: -2,147,483,648 to 2,147,483,647. 0
Node_High_Scale*	Scaling max in Connected Node	Any signed 32 bit integer in the range: -2,147,483,648 to 2,147,483,647. 100

4.4.3. Timing Parameters

Column Title	Function	Legal Values
Scan_Interval*	Seconds per Scan	0-32000, 20

⁵ Optional only for Modicon_5digit addressing, and only if Single writes do not need to be forced

4.4.4. Map Descriptor Example.

Map_Descriptor_Name	Data_Array_Name	Data_Array_Offset	Function	Node_Name	Data_Type	Address	Length	Scan_Interval
// Client Side Map Descriptors								
// Note: All three examples below are addressing the same Modbus registers.								
// For Nodes where Address_Type is ADU								
Map_Descriptors								
CMD_AI_01,	DA_AI_01,	0,	RDBC,	MODBUS_DEVICE1,	Input_Register,	1,	20,	1.000s
CMD_AO_01,	DA_AO_01,	0,	RDBC,	MODBUS_DEVICE1,	Holding_Register,	1,	20,	1.000s
CMD_DI_01,	DA_DI_01,	0,	RDBC,	MODBUS_DEVICE1,	Discrete_Input,	1,	20,	1.000s
CMD_DO_01,	DA_DO_01,	0,	RDBC,	MODBUS_DEVICE1,	Coil,	1,	20,	1.000s
// For Nodes where Address_Type is PDU								
Map_Descriptors								
CMD_AI_02,	DA_AI_02,	0,	RDBC,	MODBUS_DEVICE2,	FC04,	0,	20,	1.000s
CMD_AO_02,	DA_AO_02,	0,	RDBC,	MODBUS_DEVICE2,	FC03,	0,	20,	1.000s
CMD_DI_02,	DA_DI_02,	0,	RDBC,	MODBUS_DEVICE2,	FC02,	0,	20,	1.000s
CMD_DO_02,	DA_DO_02,	0,	RDBC,	MODBUS_DEVICE2,	FC01,	0,	20,	1.000s
// For Nodes where Address_Type is Modicon_5digit								
Map_Descriptors								
CMD_AI_03,	DA_AI_03,	0,	RDBC,	MODBUS_DEVICE3,		30001,	20,	1.000s
CMD_AO_03,	DA_AO_03,	0,	RDBC,	MODBUS_DEVICE3,		40001,	20,	1.000s
CMD_DI_03,	DA_DI_03,	0,	RDBC,	MODBUS_DEVICE3,		10001,	20,	1.000s
CMD_DO_03,	DA_DO_03,	0,	RDBC,	MODBUS_DEVICE3,		00001,	20,	1.000s

5. Configuring the FieldServer as a Modbus/TCP Server

For a detailed discussion on FieldServer configuration, please refer to the FieldServer Configuration Manual. The information that follows describes how to expand upon the factory defaults provided in the configuration files included with the FieldServer (See “.csv” sample files provided with the FieldServer).

This section documents and describes the parameters necessary for configuring the FieldServer to communicate with a Modbus TCPI Client.

The configuration file tells the FieldServer about its interfaces, and the routing of data required. In order to enable the FieldServer for Modbus TCP communications, the driver independent FieldServer buffers need to be declared in the “Data Arrays” section, the FieldServer virtual Node(s) needs to be declared in the “Server Side Nodes” section, and the data to be provided to the clients needs to be mapped in the “Server Side Map Descriptors” section. Details on how to do this can be found below.

Note that in the tables, * indicates an optional parameter, with the **bold** legal value being the default.

5.1. Server Side Connection Descriptors

Section Title		
Connections		
Column Title	Function	Legal Values
Adapter	Specify which adapter this protocol uses.	N1
Protocol	Specify protocol used	Modbus/TCP

Example:

```
// Server Side Connections

Connections
Adapter,          Protocol
N1,              Modbus/TCP
```

5.2. Server Side Node Descriptors

Section Title		
Nodes		
Column Title	Function	Legal Values
Node_Name	Provide name for node	Up to 32 alphanumeric characters
Node_ID	Node ID of physical server node	1 – 255 (Optional)
Protocol	Specify protocol used	Modbus/TCP
Address_Type ⁶	Specify Register Mapping Model	ADU,PDU, -, Modicon_5digit

Note that for this protocol, the IP address for the FieldServer is configured using the "I" menu option on the Remote User Interface.

⁶ Optional for Modicon 5 digit devices

Example

```
// Server Side Nodes
// For devices where 65536 addresses are available in each memory area.
Nodes
Node_Name,          Node_ID,          Protocol          Address_Type
MB_Srv_11,         11,              Modbus/TCP       ADU
MB_Srv_12,         12,              Modbus/TCP       PDU
// For devices where only 9999 registers are available in each memory area.
Nodes
MB_Srv_13,         13,              Modbus/TCP       Modicon_5digit
MB_Srv_14,         14,              Modbus/TCP       -
```

5.3. Server Side Map Descriptors

5.3.1. FieldServer Specific Map Descriptor Parameters

Column Title	Function	Legal Values
Map_Descriptor_Name	Name of this Map Descriptor	Up to 32 alphanumeric characters
Data_Array_Name	Name of Data Array where data is to be stored in the FieldServer	One of the Data Array names from "Setup Data Array" section above
Data_Array_Offset	Starting location in Data Array	0 to maximum specified in "Setup Data Array" section above
Function	Function of Client Map Descriptor	Server

5.3.2. Driver Specific Map Descriptor Parameters

Column Title	Function	Legal Values
Node_Name	Name of Node this Map description is associated with	One of the node names specified in "Node_Name" above
Data_Type ⁷	Specify memory area	Address_Type = ADU Coil, Discrete_Input, Input_Register, Holding_Register, Single_Coil, Single_Register Address_Type = PDU FC01, FC02, FC03, FC04, FC05, FC06, FC15, FC16 Address_Type = Modicon_5digit - (Dash), Single_Register, Single_Coil
Length	Length of Map Descriptor	Address_Type = ADU 1-65535 Address_Type = PDU 1-65535 Address_Type = Modicon_5digit 1-9999
Address	Starting address of read block	Address_Type = ADU 1-65536 Address_Type = PDU 0-65535 Address_Type = Modicon_5digit 40001, 30001, etc
Data_Array_Low_Scale*	Scaling zero in Data Array	Any signed 32 bit integer in the range: -2,147,483,648 to 2,147,483,647, 0
Data_Array_High_Scale*	Scaling max in Data Array	Any signed 32 bit integer in the range: -2,147,483,648 to 2,147,483,647, 100
Node_Low_Scale*	Scaling zero in Connected Node	Any signed 32 bit integer in the range: -2,147,483,648 to 2,147,483,647, 0
Node_High_Scale*	Scaling max in Connected Node	Any signed 32 bit integer in the range: -2,147,483,648 to 2,147,483,647, 100

⁷ Optional only for Modicon_5digit addressing, and only if Single writes do not need to be forced

5.3.3. Map Descriptor Examples

```
// Server Side Map Descriptors where Node Address_Type is ADU
// Note: All three examples below are addressing the same Modbus registers.
// For Nodes where Address_Type is ADU
Map_Descriptors
SMD_AI_01, DA_AI_01, Data_Array_Name DA_AI_01, Data_Array_Offset 0, Function Server, Node_name MB_Srv_11, Data_Type Input_Register, Address 1, Length 200, Data_Array_Low_Scale 0, Data_Array_High_Scale 100, Node_Low_Scale 0, Node_High_Scale 10000
SMD_AO_01, DA_AO_01, Data_Array_Name DA_AO_01, Data_Array_Offset 0, Function Server, Node_name MB_Srv_11, Data_Type Holding_Register, Address 1, Length 200, Data_Array_Low_Scale 0, Data_Array_High_Scale 100, Node_Low_Scale 0, Node_High_Scale 10000
```

```
// Server Side Map Descriptors where Node Address_Type is PDU
Map_Descriptors
SMD_AI_02, DA_AI_02, Data_Array_Name DA_AI_02, Data_Array_Offset 0, Function Server, Node_name MB_Srv_12, Data_Type FC04, Address 0, Length 200, Data_Array_Low_Scale 0, Data_Array_High_Scale 100, Node_Low_Scale 0, Node_High_Scale 10000
SMD_AO_02, DA_AO_02, Data_Array_Name DA_AO_02, Data_Array_Offset 0, Function Server, Node_name MB_Srv_12, Data_Type FC03, Address 0, Length 200, Data_Array_Low_Scale 0, Data_Array_High_Scale 100, Node_Low_Scale 0, Node_High_Scale 10000
```

```
// For Nodes where Address_Type is Modicon_5digit.
Map_Descriptors
SMD_AI_01, DA_AI_01, Data_Array_Name DA_AI_01, Data_Array_Offset 0, Function Server, Node_name MBP_Srv_13, Data_Type MBP_Srv_13, Address 30001, Length 200, Data_Array_Low_Scale 100, Data_Array_High_Scale 0, Node_Low_Scale 0, Node_High_Scale 10000
SMD_AO_01, DA_AO_01, Data_Array_Name DA_AO_01, Data_Array_Offset 0, Function Server, Node_name MBP_Srv_13, Data_Type MBP_Srv_13, Address 40001, Length 200, Data_Array_Low_Scale 100, Data_Array_High_Scale 0, Node_Low_Scale 0, Node_High_Scale 10000
```

Appendix A. Advanced Topics

Appendix A.1. Data Types

If Node parameter Address_Type is set as ADU or PDU, then Data_Type must be specified as follows

For Address_Type ADU :

Address range	Data_Type	Function Code (Write)	Function Code (Read)
1 - 65536	Coil	15	1
1 - 65536	Discrete_Input	n/a.	2
1 - 65536	Input_Register	n/a.	4
1 - 65536	Holding_Register	16	3

For Address_Type PDU :

Address range	Data_Type	Function Code (Write)	Function Code (Read)
0 - 65535	FC01	15	1
0 - 65535	FC02	n/a.	2
0 - 65535	FC04	n/a.	4
0 - 65535	FC03	16	3

For Address_Type Modicon_5digit

When a Modbus address range is specified, a particular Data Type is implied. The defaults are as follows:

Address range	Data_Type	Function Code (Write)	Function Code (Read)
40001 - 49999	Register	16	3
30001 - 39999	Analog_Input	n/a.	4
10001 - 19999	Digital_Input	n/a.	2
00001 - 09999	Coil	15	1

Appendix A.2. Single Writes

For pure write operations where the function = WRBC or WRBX, the driver defaults to using Function Codes 15 and 16 (Multiple writes). It is possible to force the driver to use Function Codes 5 and 6 (Single Writes) by manipulating the Data_Type parameter as follows:

For Address_Type ADU:

Address range	Data_Type	Function Code (Write)
1 - 65536	Single_Coil	5
1 - 65536	Single_Register I	6

For Address_Type PDU:

Address range	Data_Type	Function Code (Write)
0 - 65535	FC05	5
0 - 65535	FC06	6

For Address_Type Modicon_5digit

Address range	Data_Type	Function Code (Write)
40001 - 49999	Single_Register	6
30001 - 39999	Single_Coil	5.

Example: FC 6 = Write Single Register

Add a parameter to the Modbus client side Map Descriptor called Data_Type.

If you specify the Data_Type as Single_Register and the Function as WRBC or WRBX, then a Modbus poll with FC 6 will be generated.

Logically Single Register implies a length of one, and even if you try to set the length longer in the csv file, the length is limited to 1 in the driver.

Appendix A.3. Read/write Operation

When using the driver as a Modbus master, the function RDBC allows read/write capability with the Register and Coil data types. If defaults are used, then Function codes 5 and 6 (Single Writes) are used to write data back to the registers being read, regardless of data length being read. If multiple writes (FC 15 and 16) are needed for Read/write operation, then the user needs to specify the Node_Type parameter in the Client Side Nodes Section and set it to Block_Mode.

Appendix B.Driver Error Messages

Message	Description/Action
MB_TCP:#01 FYI. Server response extra bytes ignored. Cnt=%d %x	This message is printed when the TCP frame contains more bytes than a single Modbus_TCP message but insufficient extra bytes to form a second complete Modbus message. There is no explanation for the 'padding' bytes, but since the Driver ignores these extra bytes and processes the complete message correctly and you can safely ignore this message after you have completed your own sanity check. The driver prints this message once. It is suppressed on subsequent occurrences.
MB_TCP:#02 FYI. Master poll extra bytes ignored. Cnt=%d %x	
MB_TCP:#03 Err. TCP Frame has multiple MB_TCP messages. Ignored 2nd	The driver has detected enough bytes in the TCP frame for two complete Modbus_TCP messages. The second message is ignored. If this is a problem, re-configure the remote node so that only one Modbus_TCP message is contained in a single TCP frame. The driver prints this message once. It is suppressed on subsequent occurrences
